第3章 SerialLite II 活用チュートリアル

～ Altera社のFPGA間通信プロトコルを使いこなす

山田 一

ここでは，FPGA間インターフェースで用いるデータ・リンク層通信プロトコルの一つである米国Altera社の「SerialLite II」の活用法を解説する．特にSerialLite IIモジュールの生成方法について詳しく説明する．（編集部）

昨今のFPGAでは，高速シリアル伝送機能が標準化されており，どのメーカも高速シリアル伝送に対応したデバイスを出荷しています．当然ながら，これらのデバイスは高速伝送に必要な特性などを満たし，そして同時に多様なプロトコルもサポートしています．

しかし，これらのプロトコルは特定の用途を想定して策定されています．「とにかく大量のデータを伝送したい！」というかなり大ざっぱな要求に対しては，余計なオーバヘッドなどもあり，最適なプロトコルを選ぶためにいつも頭を悩ませていました．

本稿では，米国Altera社が開発した汎用性の高いプロトコル「SerialLite II」を筆者が使用した経験を基に，モジュールの生成方法を中心に解説します．

1．SerialLite IIの使いどころ

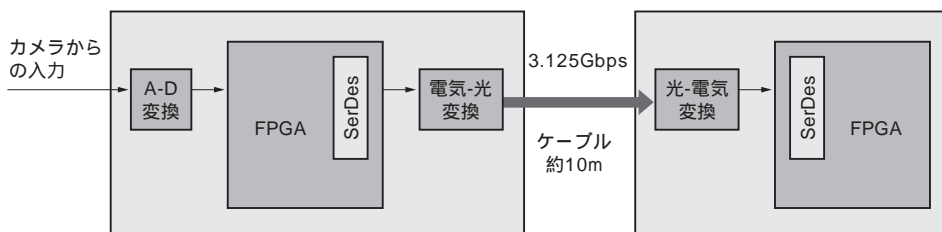
筆者が設計したシステムのブロック図を**図1**に示します．カメラから取得した大量の画像データを，後段のボードに高速に転送し，画像処理を行います．使用したFPGAは，SerDes（Serializer/Deserializer）を内蔵するAltera社の「Stratix II GX」です．

● LSI間を高速でシンプルにつなぎたい

このシステムでは，すべての回路を自分たちで設計します．市販のボードを追加することもありません．従って，ボード間のインターフェースには，どのようなプロトコルを使っても構いません．無理に業界標準のプロトコルを使う必要はありません．特定のLSI間で所望の通信が実現できればよいので，なるべく簡単にオーバヘッドの少ないプロトコルを使うのが理想です．

図1のシステムで要求される仕様は，次の通りです．

- 1チャネル構成
- 単方向の伝送

図1 FPGA間で大容量のデータを高速に通信するシステムの例
カメラから取得した大量の画像データを，後段のボードに高速に転送し，画像処理を行う．

KeyWord　FPGA，高速シリアル伝送，SerialLite II，Stratix II GX，SerDes，MegaWizard

- 部品点数を抑えたい（なるべくなら1チップで）
- 簡単に制御したい
- 回路規模を抑えたい

このような場面で活用できるのがAltera社の「SerialLite II」です．SerialLite IIは，2003年に発表されたSerialLiteをベースに改良されたものです．仕様は公開されており，幅広いアプリケーションに適用できるプロトコルになっています．

● SerialLite IIの特徴

SerialLite IIの物理層は，以下のような機能を備えています．

- 8b/10b符号化（エンコーダ/デコーダ）
- XAUIに基づいた電気的仕様
- 自動リンク確定制御

また，リンク層には，以下のような機能があります．

- 制限のないデータ・パケット・サイズ
- CRC（cyclic redundancy code）を使用したパケット転送

詳しくはSerialLite IIのWebサイト（**図2**）や仕様書を参照してください．

図2　SerialLite IIのWebサイト

URLは，「http://www.altera.co.jp/products/ip/iup/seriallite/m-alt-seriallite2.html」．

● Stratix II GXのSerDesブロック

Stratix II GXのSerDes部のブロック図（ALT2GXB）を**図3**に示します．

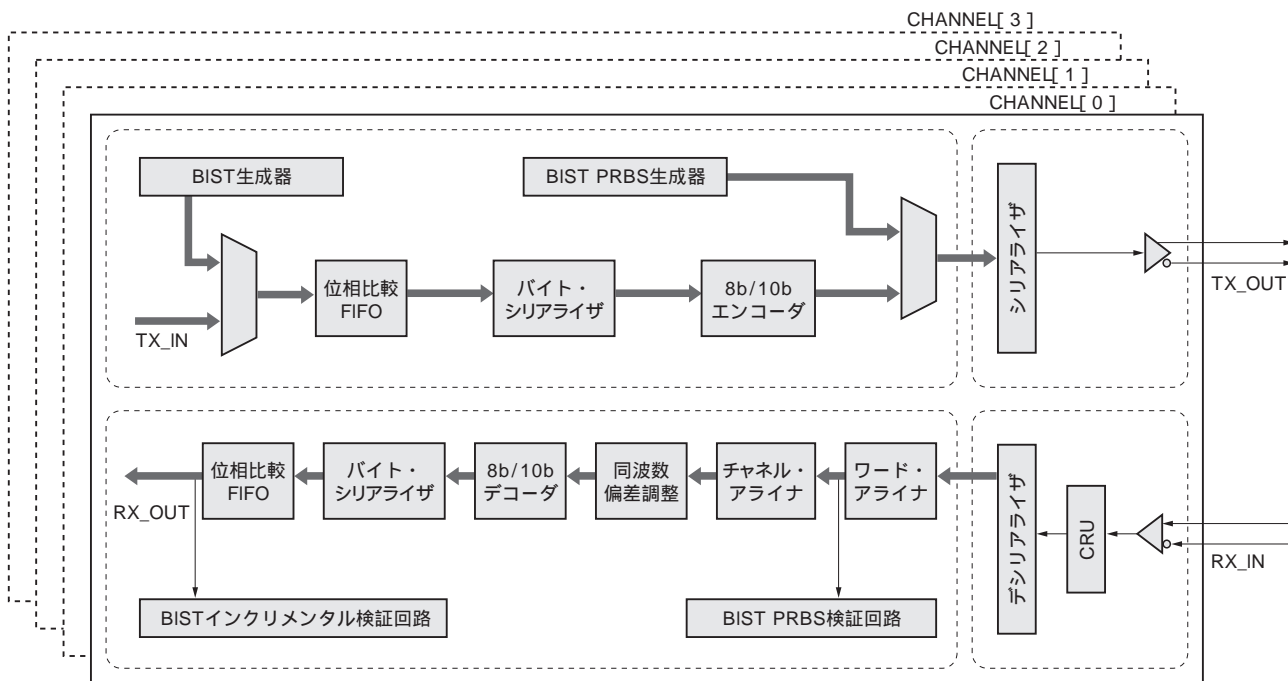



図3　Stratix II GXのSerDes部のブロック図

パラレル-シリアルの変換器や8b/10bエンコーダ/デコーダ，パラレル・データを適切なワードに調整するワード・アライナ（word aligner），チャネル間のスキューを調整するチャネル・アライナ（channel aligner），再生したクロックとリファレンス・クロックの周波数偏差調整（rate matcher）などで構成されている．

Stratix Ⅱ GXはパラレル-シリアルの変換器や8b/10bエンコーダ/デコーダ，パラレル・データを適切なワードに調整するワード・アライナ(word aligner)，チャネル間のスキューを調整するチャネル・アライナ(channel aligner)，再生したクロックとリファレンス・クロックの周波数偏差調整(rate matcher)など，高速伝送のインターフェースに必要な機能をハードウェアで持っています．

このSerDesブロックは，通常はAltera社のFPGA開発ツール「Quartus Ⅱ」を用いて構成します．しかし，上記の構成を自分で設定するとなると，意味が理解できなかったり，必要なのかどうかを判断できなかったりしてしまい，設計で悩むことがあります．

このようなとき，SerialLite Ⅱを活用すると，面倒で不可解な設定をすることなく，GUIベースのツールからモジュールを生成できます．

2. SerialLite Ⅱモジュールの生成

SerialLite Ⅱは，Quartus Ⅱをインストールすると，標準でアドインされるIPコアの一つです．わざわざ追加イン

IP tool Benchの立ち上げ画面へ

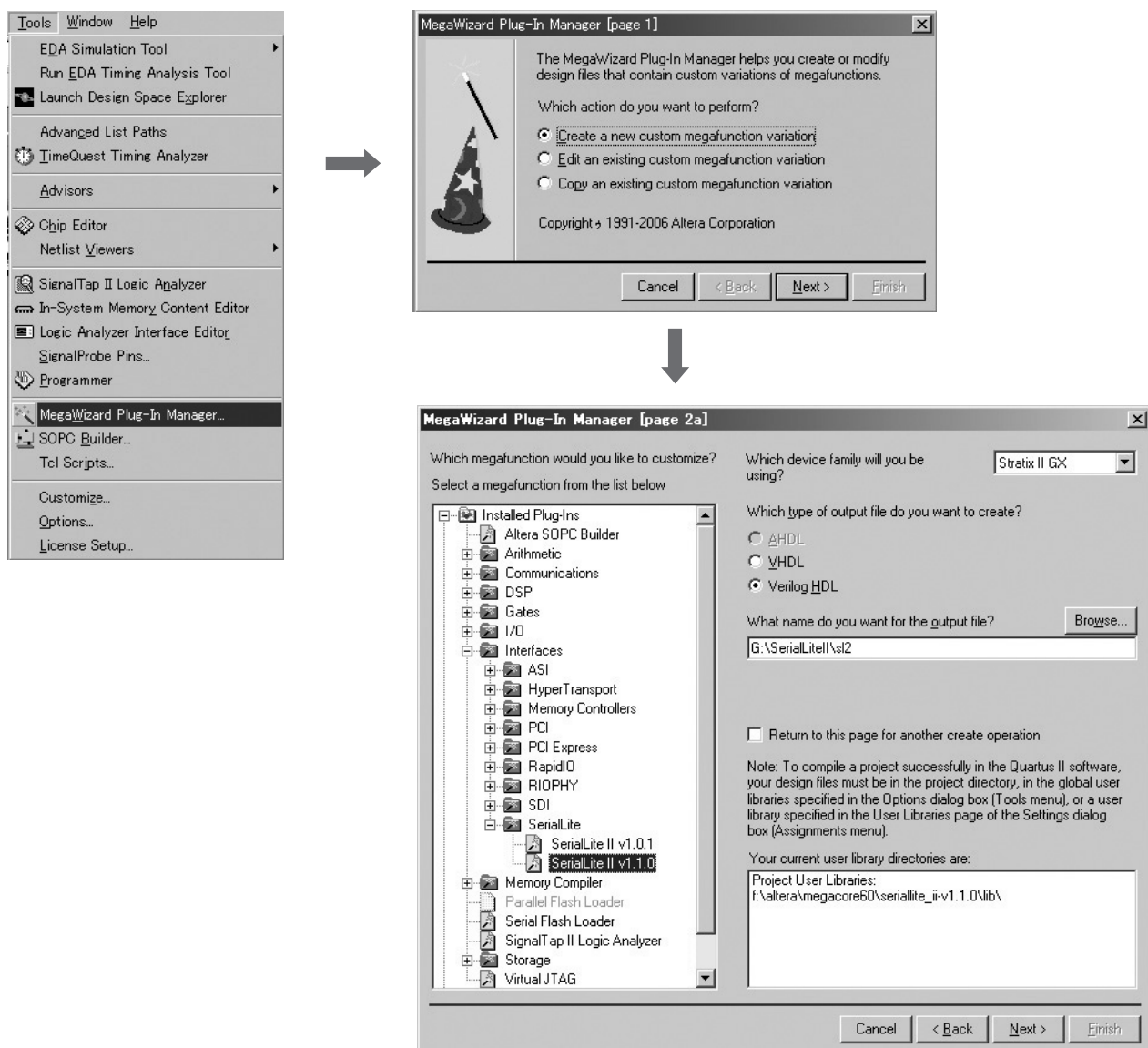



図4
SerialLite Ⅱモジュールの新規作成
MegaWizard Plug-In Managerを使用する．

ストールを行う必要はありません．対応のFPGAファミリは「Stratix II GX」と「Stratix GX」です．このため，Quartus II Web Editionでは使用することができません．

● SerialLite IIモジュールの新規作成

SerialLite IIを使用するには，MegaWizard Plug-In Managerを使います．Quartus IIでメニューから「Tool」→「MegaWizard Plug-In Manager」を選択します（**図4**）．

最初の画面では，IPコアを作成するかどうかを聞いてきます．新規に作成するときには，「Create a new custom megafunction variation」をチェックして［Next>］ボタンをクリックします．

次の画面では，左側のウィンドウに，さまざまなIPコアがツリー上に並んでいます．この中から，「Installed Plug-Ins」-「Interfaces」-「SerialLite」-「SerialLite II v1.1.0」を選択します．また，画面の右側では，ターゲットFPGAや使用する設計言語，作成するモジュール名など入力します．

［Next>］ボタンをクリックすると，IP Toolbenchが開きます（**図5**）．ここでSerialLite IIの設定を行います．

図5　IP Toolbenchの立ち上げ画面

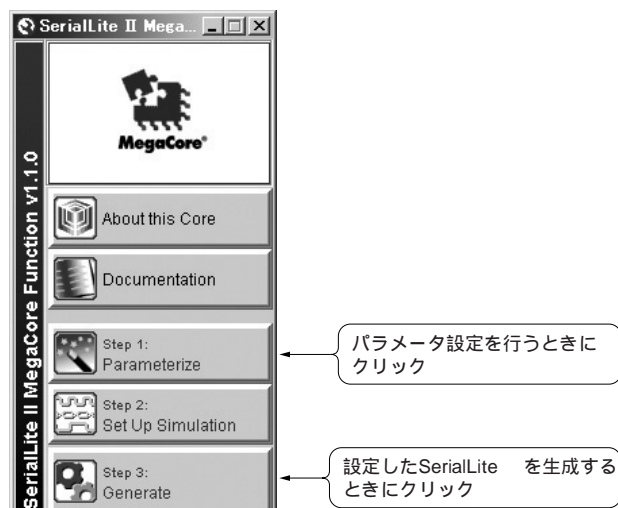
● SerialLite IIのパラメータ設定

図6はSerialLite IIのIPコア設定画面です．

1）物理層の設定

「Physical Layer」タブでは，SerialLite IIの物理層に関する項目を設定します．

現在，SerialLite IIに対応しているFPGAはStratix GXとStratix II GXです．従って，Device family欄では，いずれかを選択します．ここではStratix II GXを選択します．

Data SettingsのData rate欄には，データ伝送速度を入力します．単位はMbpsなので，3.125Gbpsなら「3125」とします．Stratix II GXは最大6.375Gbpsのデータ伝送速度に対応できます．Transfer size欄では，パラレル・データ長を設定します．通常，シリアルで送受信するGbpsオーダのデータ・レートのままでは，ユーザ論理を構成するFPGAの基本論理ブロックは動作しません．このためシリアル-パラレル変換回路があります．このパラレル・データの幅がTransfer sizeです．単位はColumnsで，設定値は1，2，4になります．1 Columnsの場合は8ビット，2 Columnsの場合は16ビットになります．これはSerialLite IIは8b/10b符号化を必ず使用するため，パラレル・データ幅は8ビット単位になるからです．ビット数を増やせば，ユーザ論理の動作周波数を下げられる可能性が出てきます．ユーザ論理の動作周波数をどの程度にするかによって設定値を決めるとよいでしょう．

Port Typeでは，SerialLite IIの通信方向を選択します．双方向（Bidirectional），送信だけ（Transmitter only），受信だけ（Receiver only）のいずれかを選べます．双方向にした場合は，送受信の速度が同じになります．また，通常，送信側は受信側が正常に動作している前提でデータの送信を行いますが，必ずしも受信側が正常に動作しているとは限りません．そこで受信側が正常に動作しているかどうかを確認するデータ（リンクアップ信号）を流し，受信側はそのデータを受け取ったら送信側に応答します．この一連の流れは「自動同期化リンク制御」と呼ばれ，SerialLite IIの標準機能の一つです．この「自動同期化リンク制御」を有効にするかどうかを，Self-Synchronized Link-Upのチェック欄で設定します．双方向で通信する場合は使用した方がよいでしょう．

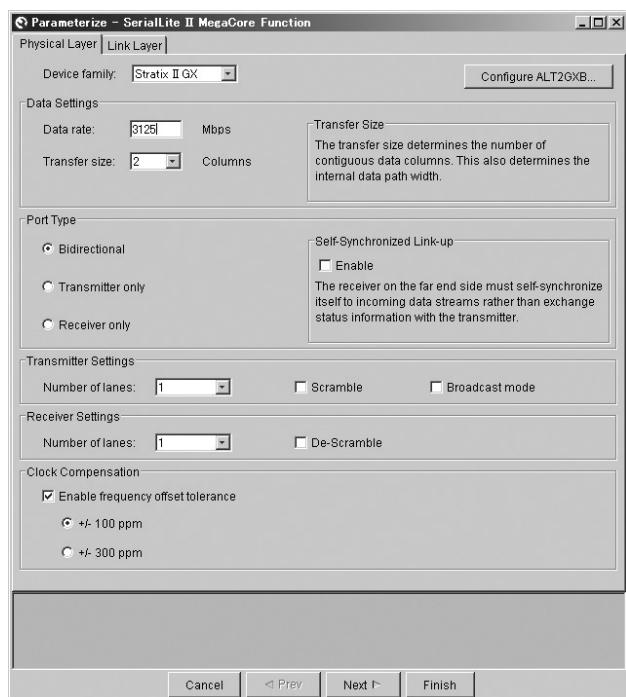
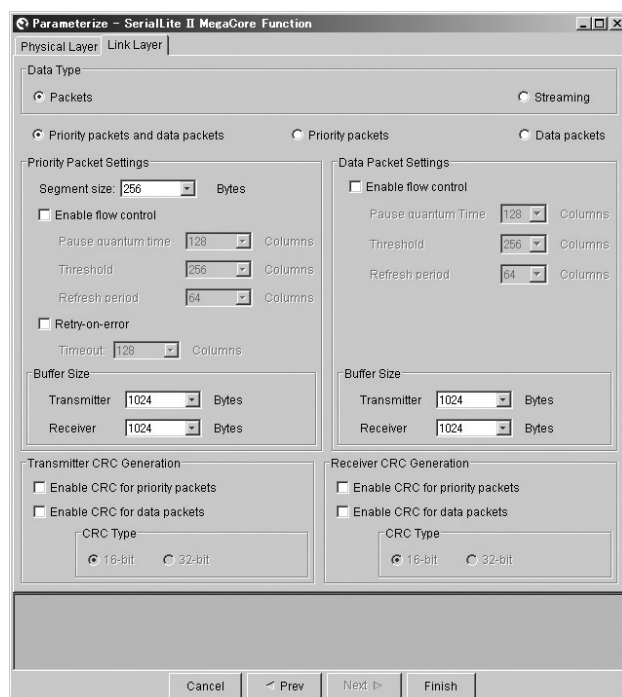
Transmitter SettingsとReceiver Settingsでは，送受信のレーン数を決めます．使用できるレーン数は，FPGAの品種により異なります．Stratix II GXでは，最大16レーンまで使用できます．また，送信時に生成多項式を用いたランダムなパターンにデータを変換する場合は，「Scramble」をチェックします． この変換によって伝送路上のEMIが向上します．Scrambleしたデータを復元する場合には，受信側で「De-Scramble」を選択する必要があります．一つのパラレル・データを複数のシリアル・ポートに分配する場

（a）物理層の設定

（b）リンク層の設定

図6　SerialLite Ⅱのパラメータ設定

物理層の設定とリンク層の設定を行う．

合には，「Broadcast mode」を用います．すると1：nの伝送が可能になります．

Clock Compensationは，周波数偏差を吸収する機能の設定です．Stratix Ⅱ GXは受信したデータからクロックを再生して受信ブロックのクロックとして使用できます（CDR：clock data recovery）．ただし通信の途中で，リファレンス・クロックに乗せ換えます．この再生したクロックとリファレンス・クロックには，周波数に偏差があるので，ある決まったタイミングでは，どうしてもクロックの乗せ換えができない瞬間が発生してしまいます．この現象を防ぐためには，定期的に周波数偏差を吸収する必要があります．この偏差を吸収するパターンを送出するモードが「Clock Compensation」の設定です．この偏差の数値として，±100ppmもしくは±300ppmを選ぶことにより，偏差吸収パターンの送出頻度が変わります．

2）リンク層の設定

「Link Layer」タブではリンク層の設定を行います．

SerialLite Ⅱでは，データの送出方法として，パケット方式とストリーミング方式の2種類に対応します．これをData Typeで選択します．また，パケットにも2種類あります．通常はデータ・パケット（Data packets）が送出されますが，優先度の高いプライオリティ・パケット（Priority packets）も使用できます．これをData Typeの下のチェックで選択します．

Priority Packet SettingsとData Packet Settingsで，パケットの設定を行います．Priority Packetの場合は，Segment sizeでパケット長を指定します．最大2048バイトまで設定可能です． Data Packetにはパケット長の規定はありません．

SerialLite Ⅱのモジュールからユーザ論理に渡すパスにはFIFOが配置されています．しかしユーザ論理の設計によっては，このFIFOからのデータを受け取りきれない状態になり，FIFOがオーバフローを起こしてしまう可能性があります．そのため受信側FIFOの状態を送信側に送るフロー制御（flow control）の機能があります．このフロー制御を使用するかどうかの指定をEnable flow controlで行います．FIFOがどのような状態になったときに送信側からの転送を中断するかを決められます． SerialLite Ⅱのモジュールからユーザ論理に渡すパスにあるFIFOのサイズは，Buffer Sizeで設定します．

受信したデータが正しくなかった場合に，送信側へ同じデータの再送信を要求する機能として，Retry-on-errorが

あります．この機能を使用するかどうかの指定をRetry-on-errorのチェックで行います．

Transmitter/Receiver CRC Generationでは，パケットにCRC符号を付加するかどうかを指定します．

図1のシステムの設定例を，**表1**にまとめます．

表1　SerialLite Ⅱのパラメータ設定例

Physical Layerタブ	
Device family	Stratix II GX
Data rate	3125 Mbps
Transfer size	2 Columns
Port type	Bidirectional
Self-Synchronized Link-up	Enable
Number of lanes（Transmitter & Receiver）	1
Scramble/De-Scramble	Disable
Broadcast mode	Disable
Frequency offset tolerance	Enable
Link Layerタブ	
Data type	Packets
Packet type	Data packets
Flow control	Disabled
Buffer size（Transmitter & Receiver）	1,024 Bytes
CRC generation	Disabled

3．デバッグ時のポイント

ところで，実際に高速シリアル伝送をFPGAで実現する場合，SerialLite ⅡのようなFPGAベンダが提供するIPコアなどを使えば比較的簡単に実現できます．前述したように，各種設定もGUI画面でパラメータを与えられるので，便利です．

しかし実際のボード・デバッグは容易ではありません．Gbpsの信号を観測するにしても高価で高精度の計測器が必要です．最近ではレンタルというケースも多いのですが，この場合は使用できる時間が限られてしまいます．

効率良くデバッグするためには，あらかじめ設計段階において，各機能の動きを理解しておくことが重要なポイントとなります（例えば8b/10bなど）．これを怠ると，いざ動かなかった場合に何が悪いのか切り分けができません．

伝送路を疑って，時間をかけて計測器でいろいろ調べていたのに，実はFPGAの中のIPコアの設定が間違っていたこともありました．またそれが以前に自分で設計した論理のバグだったこともありました．まず実機で確認する前にFPGAの開発時に検証できるところはしっかりと検証することが重要です．そして，何が正常動作なのかをしっかり把握してから実機調整に臨まないと，デバッグに膨大な時間がかかってしまいます．

また，設計時にデバックしやすい工夫を入れたりしておくことが大切です．最近ではFPGA内部の信号を観測できるツール（Altera社の場合ならSignalTap Ⅱ）があります．あらかじめポイントになる部分を検証できるように考慮しておくとよいでしょう．テスト・パターンなども有効に使えるようにしておくとさらに安心です．

実機でテストする前の準備が非常に大切だということです．

初めて高速シリアル伝送での開発をする場合は，FPGAベンダが提供している開発ボードを参考に設計するとよいでしょう．特にパターン設計や電源周りはいきなり設計するのは敷居が高いため，かなり参考になると思います．

やまだ・はじめ